LLNL-CONF-568312

# Exploring Performance Data with Boxfish

K. Isaacs, A. Landge, T. Gamblin, T. Bremer, B. Pascucci, B. Hamann

August 1, 2012

Supercomputing 2012
Salt Lake, UT, United States
November 10, 2012 through November 16, 2012

# Exploring Performance Data with Boxfish

Katherine E. Isaacs*, Aaditya Landge†, Todd Gamblin‡, Peer-Timo Bremer‡, Valerio Pascucci†, Bernd Hamann*

‡Center for Applied Scientific Computing, Lawrence Livermore National Laboratory, Livermore, CA 94551 USA
†Scientific Computing and Imaging Institute, The University of Utah, Salt Lake City, UT 84112 USA
*Department of Computer Science, University of California, Davis, CA 95616 USA
E-mail: keisaaacs@ucdavis.edu, {aadtiya, pascucci}@sci.utah.edu, {tgamblin, ptbremer}@llnl.gov, hamann@cs.ucdavis.edu

The growth in size and complexity of scaling applications and the systems on which they run pose challenges in analyzing and improving overall performance. With metrics coming from thousands to millions of processes, visualization techniques are necessary to make sense of the increasing amount of data. To aid the process of exploration and understanding, we announce the initial release of Boxfish, an extensible tool for manipulating and visualizing data pertaining to application behavior. Combining and visually presenting data and knowledge from multiple domains, such as the application's communication pattern and hardware's network configuration and routing policies, can yield the insight necessary to discover the underlying causes of observed behavior. Boxfish allows users to query, filter and project across these domains to create interactive, linked visualizations.

## I. Projecting Data Across Domains

We describe the association of elements that exist in one domain with the elements of another as a *projection*. A map file, which associates integer MPI ranks with coordinate-denoted hardware nodes and threads, is an example of a commonly used projection. Schulz et al. [1] advocated the use of projections in interpretting performance data and defined three domains of interest – hardware, application and communication. The hardware domain includes performance counters. The application domain includes information relating to the application, such as physics measurements in a simulation or matrix properties in a linear algebra library. The communication domain includes messages sent between processes and communication sets of nodes such as MPI communicators. Boxfish recognizes these domains. Contributed modules may add others.

Boxfish is designed to support projecting data across domains. When filters or queries are written requiring attributes from multiple domains, or when a view requires attribute information in a native domain, Boxfish which search its available projections to make the necessary transformations. This allows users to view data such as the load on nodes that had a range of values in previous run or the average wait time for communicators in a particular phase. Data tables corresponding to runs may have default preferred projections. Projections can be added from files, created based on data attributes, or composed from existing ones. More complex projections may be added through future or contributed modules.

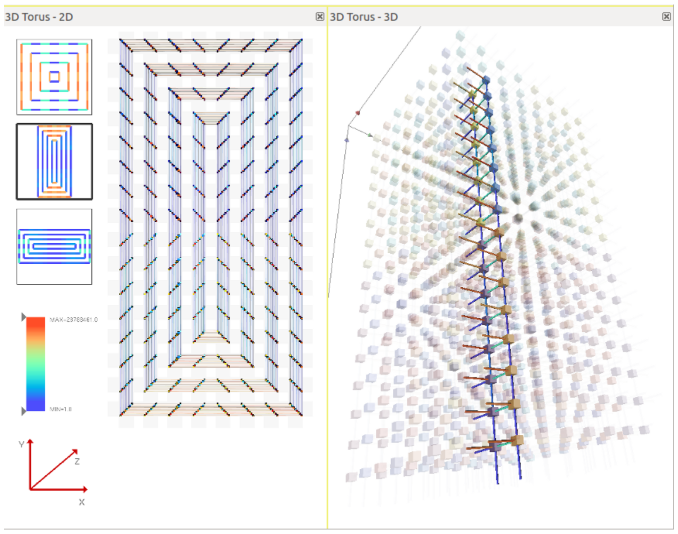

Fig. 1. A 3D torus represented in 2D (left) and 3D (right). Both views represent elements of the hardware domain. However, nodesare colored by their MPI sub-communicators. Links are colored by traffic. These views are side by side in Boxfish, indicating they are siblings in the filter hierarchy, indicating they show the same data. In the 2D view, selected nodes are represented by larger size. In the 3D view, the same nodes are selected, represented by opacity.

Figure 1 displays a projection from the communication domain onto the hardware domain. The nodes of the hardware domain are colored by the values of the MPI sub-communicators in the communication domain. This particular case is a one-to-many projection, where a single sub-communicator maps to many nodes. Boxfish also handles many-to-one projections through a variety of data combining functions.

## II. Filter Hierarchy in Boxfish

Inspired by the data flow in Epinome [2], Boxfish allows filters and views to be grouped hierarchically. This allows both the data manipulation and the view interactions to be applied simultaneously to several views at once. Users can move views and groups of views anywhere in the hierarchy, dynamically changing the presented data.

When elements are selected in one view, corresponding elements in other views may be automatically highlighted depending on their position in the hierarchy and the policies of each subtree. Though the elements shown in each view may

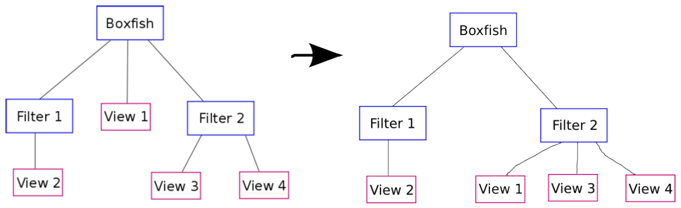

Fig. 2. Two potential Boxfish configurations. On the left, the user has four views, two of which share a filter on their information and one of which has no filter. On the right, the user has moved View 1 under Filter 2, so its data matches that of View 3 and View 4.

not be the same, selection in one can induce selection in the others if the first set of elements can be projected onto the others.

## III. Boxfish Views

Boxfish is designed to facilitate the addition of new views by unifying the processing and manipulating of input data and handling shared view actions such as highlighting. Details on writing a Boxfish view are available with the user manual. In this section, we briefly describe the initial set of views in the initial Boxfish release.

**Plot** – Boxfish's default plotting module can create scatter plots and histograms of (potentially aggregated) data attributes. Attributes from one or multiple domains may be plotted against the natural elements of any domain for which a projection exists. The plots offers a familiar method of looking at the data and selecting features of interest which can then be highlighted in other views.

**Torus/Mesh 3D** – Three-dimensional torus and mesh networks are represented in their conventional configuration. In the case of the torus, wrap-around links are shown from one of their two end-nodes. The location of this 'seam' in any of the dimensions can be changed. Attributes are displayed on nodes or links through use of color. This view is shown in Figure 1.

**Torus/Mesh 2D** – Three-dimensional torus and mesh networks are arranged on a 2D plane to eliminate occlusion and ease selection. To achieve this representation, the links in one direction are not shown. Users can change which dimension is omitted and all three configurations are shown in overview 'minimaps' that aggregate attribute values on the links. Like its 3D counterpart, attributes are shown in this view by coloring the nodes and links. This view is shown in Figure 1.

**Communication graph** – The set of messages sent between pairs of processes can be displayed as an adjacency matrix where the number of messages or bytes sent in a single direction is depicted as a color value. The order of the processes may be altered to represent other attributes.

## IV. Boxfish in Practice

Bhatele et al. [3] used Boxfish's 3D torus view to help identify the cause of a scaling problem in SAMRAI [4], an adaptive mesh refinement library. Nodes that spent the longest in a load balancing phase appeared clustered in the 3D torus view, leading to further investigation into the cause of that effect.

Boxfish's 2D torus view has been used to understand network behavior [5] in pF3D [6], a laser-plasma interaction simulation, and QBall, a molecular dynamics simulation. The view showed the differences in traffic load in the various torus directions given various node mappings. The 3D torus view [5] was also used to verify the topological layout of particular node mappings.

## Acknowledgments

This work was performed under the auspices of the U.S. Department of Energy by Lawrence Livermore National Laboratory under Contract DE-AC52-07NA27344 (LLNL-???-######). This research was supported in part by an award from the Department of Energy (DOE) Office of Science Graduate Fellowship Program (DOE SCGF). The DOE SCGF Program was made possible in part by the American Recovery and Reinvestment Act of 2009. The DOE SCGF program is administered by the Oak Ridge Institute for Science and Education for the DOE. ORISE is managed by Oak Ridge Associated Universities (ORAU) under DOE contract number DE-AC05-06OR23100. All opinions expressed in this paper are the author's and do not necessarily reflect the policies and views of DOE, ORAU, or ORISE.

## References

[1] M. Schulz, J. Levine, P.-T. Bremer, T. Gamblin, and V. Pascucci, "Interpreting performance data across intuitive domains," in *Parallel Processing (ICPP), 2011 International Conference on*, September 2011, pp. 206–215.

[2] Y. Livnat, T.-M. Rhyne, and M. H. Samore, "Epinome: A visual-analytics workbench for epidemiology data," *IEEE Computer Graphics and Applications*, vol. 32, no. 2, pp. 89 – 95, 2012.

[3] A. Bhatele, T. Gamblin, K. E. Isaacs, B. T. N. Gunney, M. Schulz, P.-T. Bremer, and B. Hamann, "Novel views of performance data to analyze large-scale adaptive applications," in *Proceedings of the 2012 ACM/IEEE conference on Supercomputing*, ser. SC '12. IEEE Press, 2012, p. To appear.

[4] B. T. N. Gunney, "Large-scale dynamically adaptive structured AMR," SIAM Conference on Parallel Processing for Scientific Computing, February 2010, uCRL-PRES-422996.

[5] A. G. Landge, J. A. Levine, K. E. Isaacs, A. Bhatele, T. Gamblin, M. Schulz, S. H. Langer, P.-T. Bremer, and V. Pascucci, "Visualizing network traffic to understand the performance of massively parallel simulations," *IEEE Transactions on Visualization and Computer Graphics (InfoVis 2012)*, p. To appear, 2012.

[6] C. H. Still, R. L. Berger, A. B. Langdon, D. E. Hinkel, L. J. Suter, and E. A. Williams, "Filamentation and forward brillouin scatter of entire smoothed and aberrated laser beams," *Physics of Plasmas*, vol. 7, no. 5, p. 2023, 2000.